Sensing cost for automata and synthesis

Shaull Almagor\textsuperscript{1}, Denis Kuperberg\textsuperscript{2}, Orna Kupferman\textsuperscript{1}

\textsuperscript{1}Hebrew University of Jerusalem
\textsuperscript{2}TU Munich

Séminaire MOVE
25-02-2016
LIF, Marseille
- **Deterministic** automata scanning the environment and checking a specification.
- **Deterministic** automata scanning the environment and checking a specification.

- **Input:** $S$ set of signals, $\Sigma = 2^S$ alphabet of the automaton.
- Deterministic automata scanning the environment and checking a specification.

- **Input:** $S$ set of signals, $\Sigma = 2^S$ alphabet of the automaton.

- **New approach:** Reading signals via sensors costs energy.
- **Deterministic** automata scanning the environment and checking a specification.

- **Input:** \( S \) set of signals, \( \Sigma = 2^S \) alphabet of the automaton.

- **New approach:** Reading signals via sensors costs **energy**.

- **Goal:** Minimize the energy consumption in an average run.
Deterministic automaton $\mathcal{A}$ on 2 signals.

$q$ state: $scost(q) =$ number of relevant signals in $q$. 
Deterministic automaton $\mathcal{A}$ on 2 signals.

$q$ state: $scost(q) =$ number of relevant signals in $q$. 

$$
\begin{array}{c}
00, 11 & 10, 01 & 10, 11 \\
\rightarrow & \uparrow & \rightarrow \\
2 & 00, 01 & 1
\end{array}
$$
Sensing cost of a deterministic automaton

Deterministic automaton $A$ on 2 signals.

$q$ state: $scost(q) = \text{number of relevant signals in } q.$

$w$ word: $scost(w) = \text{average cost of states in the run of } A \text{ on } w.$
Sensing cost of a deterministic automaton

Deterministic automaton $\mathcal{A}$ on 2 signals.

$q$ state: $\text{scost}(q) =$ number of relevant signals in $q$.

$w$ word: $\text{scost}(w) =$ average cost of states in the run of $\mathcal{A}$ on $w$.

$$\text{scost}(\mathcal{A}) = \lim_{m \to \infty} |\Sigma|^{-m} \sum_{|w|=m} \text{scost}(w)$$

Always converge.
Computing the cost

Remarks on the definition of sensing cost:

- Initial state plays a role but not acceptance condition.
- Works on finite or infinite words.
- Cost is deduced from the transition structure.
- Signals can be weighted with different probabilities or sensing cost.
Remarks on the definition of sensing cost:

- Initial state plays a role but not acceptance condition.
- Works on finite or infinite words.
- Cost is deduced from the transition structure.
- Signals can be weighted with different probabilities or sensing cost.

**Theorem**

*Sensing cost of an automaton is computable in polynomial time.*

By computing the stationary distribution of the induced Markov chain.
Stationary distribution: $\frac{1}{2}, \frac{1}{2}$

Sensing cost: $\frac{3}{2}$. 
Stationary distribution: $\frac{2}{5}, \frac{3}{5}$

Sensing cost: $\frac{7}{5}$. 
Sensing cost of a regular language

Sensing cost as a measure of complexity of regular languages.

\[ \text{scost}(L) := \inf \{ \text{scost}(A) | L(A) = L \} . \]

Can we compute the sensing cost of a language? How hard is it?
Sensing cost of a regular language

Sensing cost as a measure of complexity of regular languages.

\[ \text{scost}(L) := \inf \{ \text{scost}(A) | L(A) = L \}. \]

Can we compute the sensing cost of a language? How hard is it?

**Theorem**

*On finite words, the optimal sensing cost of a language is always reached by its minimal automaton.*
Sensing cost of $\omega$-regular languages

- On infinite words: deterministic parity automata.
On infinite words: deterministic parity automata.

Computing the minimal number of states is \( \textbf{NP} \)-complete [Schewe ’10].
Sensing cost of $\omega$-regular languages

- On infinite words: deterministic parity automata.
- Computing the minimal number of states is $\text{NP}$-complete [Schewe ’10].
- Third complexity measure of $\omega$-languages: parity rank.
Sensing cost of $\omega$-regular languages

- On infinite words: deterministic parity automata.
- Computing the minimal number of states is $\mathbf{NP}$-complete [Schewe ’10].
- Third complexity measure of $\omega$-languages: parity rank.

**Theorem**

The sensing cost of an $\omega$-regular language is the one of its residual automaton.

**Corollary**

Computing the sensing cost of an $\omega$-regular language is in $\mathbf{P}$. 
Remarks on the result:

- Optimal sensing cost might be reached only in the limit, not by a particular automaton.
Remarks on the result:

- Optimal sensing cost might be reached only in the limit, not by a particular automaton.
Remarks on the result:

- Optimal sensing cost might be reached only in the limit, not by a particular automaton.
- Trade-off between sensing cost and size.
Remarks on the result:

- Optimal sensing cost might be reached only in the limit, not by a particular automaton.
- Trade-off between sensing cost and size.
- No trade-off between sensing cost and parity rank.
Remarks

Remarks on the result:

- Optimal sensing cost might be reached only in the limit, not by a particular automaton.
- Proof uses lemma of [Niwinski, Walukiewicz '98] on the structure of automata of optimal parity index.
- Trade-off between sensing cost and size.
- No trade-off between sensing cost and parity rank.
- Idea of the proof of general interest: one can “ignore” the input for arbitrary long periods and still recognize the language.
Limitation of the probabilistic model: Safety automata always have cost 0. Only ergodic components matter in the long run.

Solution: Average only on accepted words.
Safety Setting

Limitation of the probabilistic model: Safety automata always have cost 0. Only ergodic components matter in the long run.

Solution: Average only on accepted words.

Two options:
Word average:

\[ w\text{cost}(A) = \lim_{m \to \infty} \frac{1}{L \cap |\Sigma|^m} \sum_{w \in L \cap |\Sigma|^m} s\text{cost}(w) \]

Letter-by-letter: \( l\text{cost}(A) \) via Markov chain induced by \( A \): letters are randomly picked step-by-step to stay in \( L \).

Equivalent when all words were considered, different if we restrict attention to \( L \).
Two variants of safety cost

After 11, no more 1 on the first component.

\[ lcost(A) = 1, \ wcost(A) = 2. \]

**Remark:** \( wcost \) takes into account the transient components: if left self-loop has 2 labels, then \( wcost(A) = 3/2 \).
Two variants of safety cost

After 11, no more 1 on the first component.

\[ lcost(A) = 1, \quad wcost(A) = 2. \]

**Remark:** \( wcost \) takes into account the transient components: if left self-loop has 2 labels, then \( wcost(A) = 3/2 \).

**Theorem**

\( lcost(A) \) and \( wcost(A) \) are computable in polynomial time. Their minimal is reached on the minimal automaton.

For \( wcost \): generating series, algorithms on algebraic numbers.
Cost of synthesis of a I/O specification \( L \): Infimum of costs of transducers realizing \( L \).

**Computational problem:**

**Input:** Deterministic automaton \( D \) for \( L \subseteq (I \cup O)^\omega \).

**Output:** Cost of synthesis of \( L \).
Cost of synthesis of a I/O specification $L$: Infimum of costs of transducers realizing $L$.

**Computational problem:**

**Input:** Deterministic automaton $D$ for $L \subseteq (I \cup O)\omega$.

**Output:** Cost of synthesis of $L$.

**Theorem**

*For a safety specification, the problem is EXPTIME-complete, and an optimal transducer always exists.*

**Remarks**

- Without cost constraint: $P$ with transducer of size $|D|$
- Optimal transducer can be exponential in the input deterministic automaton.
- $\leadsto$ Membership in EXPTIME by a game argument.
- Hardness by reduction from Tree Automata Intersection.
- For general languages as inputs, decidability open.
Conclusion

Results

- General definition of sensing cost for finite and infinite words.
- Optimal cost computable in \( \text{P} \).
- Refined definitions for safety languages, well-behaved.
- \( \text{EXPTIME} \)-completeness of optimal safety synthesis.
- Minimally-sensing transducer for safety specifications (exponential)

Future work:

- Decidability of cost of synthesis for parity specifications
- Precise study of the trade-off between different complexity measures
- Refining the model: cost of switching, . . .